Ollscoil na Gaillimhe
University of Galway

CT336/CT404
Graphics & Image Processing

Sem 1 (Autumn) 2024-25,
Dr. Nazre Batool

4th year B.Sc. (CS&I.T.).
2nd year M.Sc. (Software Development & Design)
1st year M.Sc. (Biomedical Engineering)
Visiting students

University *of* Galway.ie

Lecture 4: Image Filtering

Ollscoil na Gaillimhe
University of Galway

- Last Lecture:
  - Introduction to image processing
  - Point and geometric transformations
  - Spatial filtering (part 1 of 2)
    - 2D Convolution
    - Smoothing filters

- Today:
  - Image Filtering in Spatial Domain (part 2 of 2)
    - Differentiating filters: 1st order and 2nd order
  - Image Filtering in Frequency Domain
    - What is a frequency domain? (some Digital Signal Processing!)
    - How are images represented as (2D) frequencies? 2D Fourier Transform
    - Low-pass and high-pass filters

Spatial Filtering (2D-Convolution)

Ollscoil na Gaillimhe
University of Galway

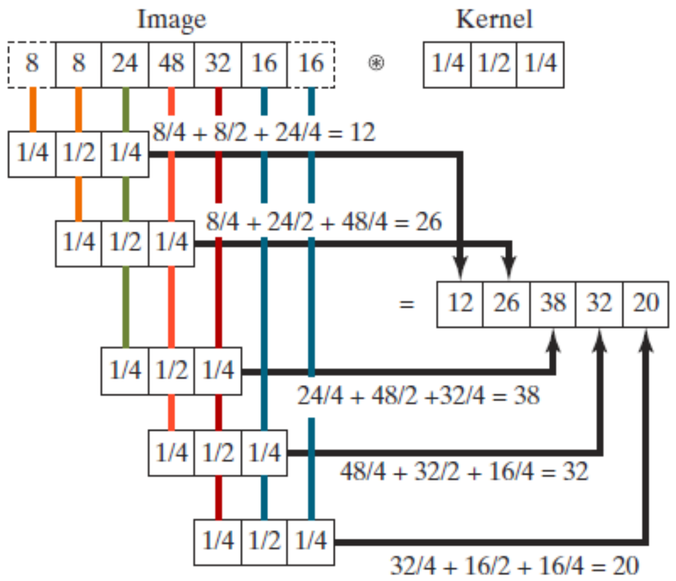


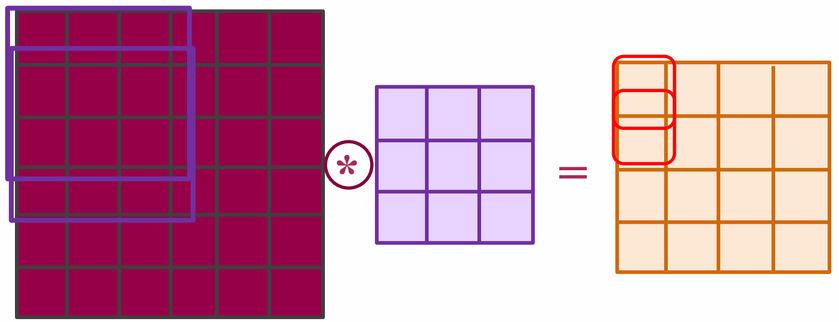
Figure 5.1 An example of 1D convolution.

- A fundamental local operation in image processing
- Used for a variety of tasks, including noise removal, blurring, sharpening, and edge detection.

- Establish a moving window ('kernel') which contains an array of coefficients or weighting factors.

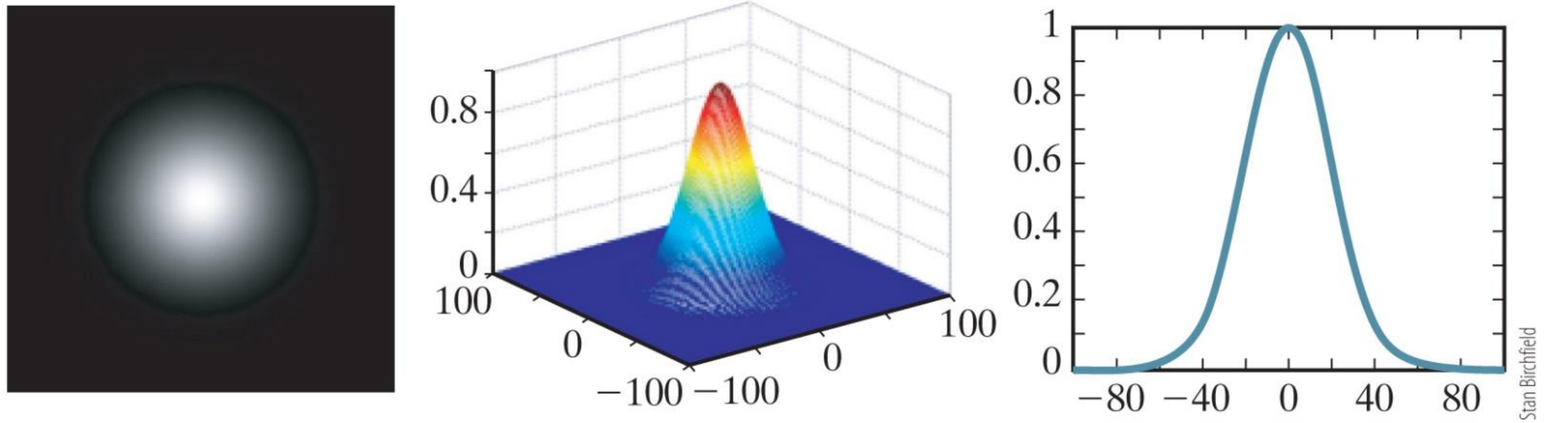
- Move the kernel across the original image, so that it centres on each pixel in turn. Each coefficient in the kernel is multiplied by the value of the pixel below it, and the addition of each of these values determines the value of the pixel in the output image corresponding to the pixel in the centre of the kernel.

- The operation of convolution is denoted by $\circledast$
- For symmetric kernels with real numbers and signals with real values (as is the case with images), convolution is the same as cross-correlation.



Gaussian Smoothing Kernel

Figure 5.3 A Gaussian is a bell curve. From left to right: The 2D isotropic Gaussian viewed as an image where the gray level of each pixel is proportional to the value of the Gaussian function at that point, the 2D isotropic Gaussian viewed as a surface in 3D, and the 1D Gaussian function (or, equivalently, a slice through the 2D Gaussian function, obtained by intersecting it with a vertical plane).



1/16

1	2	1
2	4	2
1	2	1

1/273

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

1/1003

0	0	1	2	1	0	0
0	3	13	22	13	3	0
1	13	59	97	59	13	1
2	22	97	159	97	22	2
1	13	59	97	59	13	1
0	3	13	22	13	3	0
0	0	1	2	1	0	0

Discrete Approximation of Isotropic Gaussian filter 3x3, 5x5, 7x7

Image Filtering for Noise Reduction

Typically use ‘smoothing’ *high-frequency* noise without unduly damaging larger *(low-frequency)* objects of interest

Commonly used smoothing filters.

- 1. Blur - averages a pixel and its neighbours
- 2. Median - replaces a pixel with the median, rather than the mean, of the pixel and its neighbours
- Gaussian - a filter that produces a smooth response (unlike blur/’box’ and median filtering) by weighting more towards the centre

Original

Box (Average) Filter 3x3

Gaussian filter: 3x3, sigma 0.5

7x7, sigma 0.5

7x7, sigma 1.5

Image Filtering

Ollscoil na Gaillimhe
University of Galway

Why do we need to filter images? A typical “classical” computer vision pipeline has following steps:

1. Clean up/Pre-processing
   - reduce noise (smoothing kernels)
   - remove geometric/radiometric distortion
   - emphasise desired aspects of image e.g. edges, corners, blobs, etc. (differentiating kernels, feature detectors)
2. Segmentation *(might not be needed in deep learning based approaches)*
   - identify/extract objects of interest
   - sometimes the entire image is ‘of interest’ and the task is to separate it into non-overlapping regions
   - probably leverages domain-specific knowledge
3. Measurement *(might not be needed in deep learning based approaches)*
   - quantify appropriate measurements on segmented objects
4. Classification
   - assign segmented objects to classes
   - make decisions etc.

Edge Detector/Differentiating Filters as Mathematical Derivatives

- Consider a horizontal slice across the image
- Edge detection filters are essentially performing differentiation of grey level with respect to distance

- i.e. ‘how different is a pixel to its neighbours’?

- Some filters are akin to **first derivatives**, while others are akin to **second derivatives**

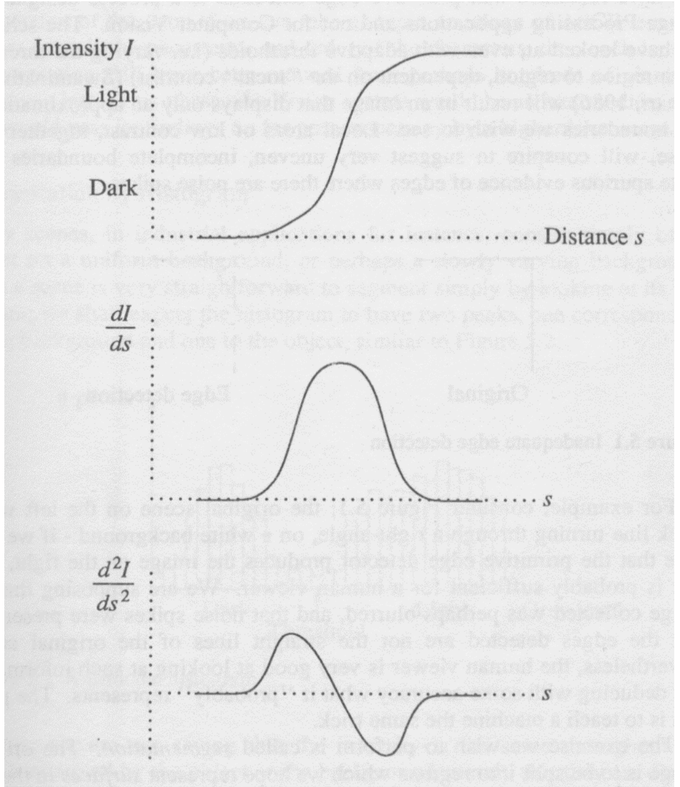


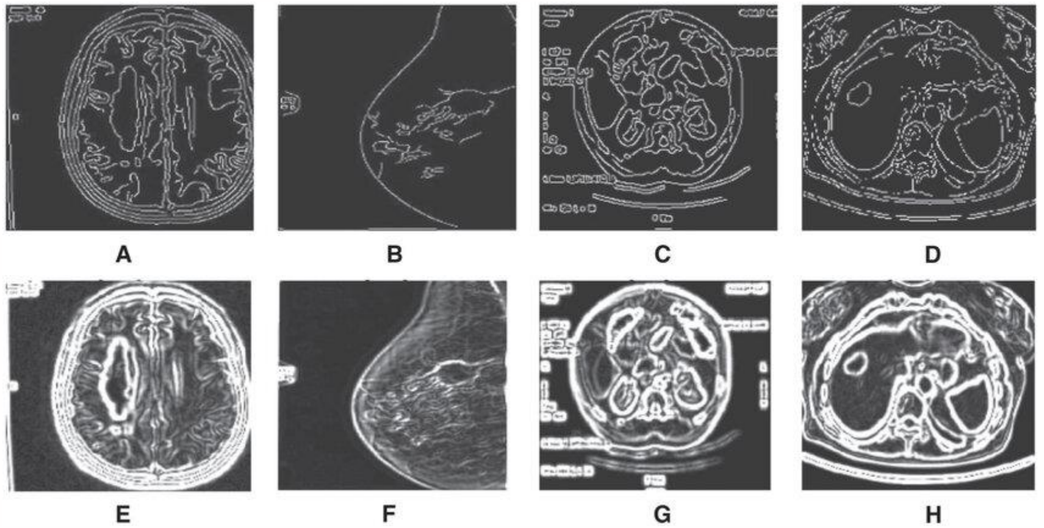
Image Filtering for Edge Detection

Differentiating kernels can represent **first order** or **second order** derivatives.

Differentiating kernels for edge detection can also be classified as **gradient magnitude** type or **gradient direction** type.

What is Edge Detection? (will cover more in a later lecture)

- A common early step in image segmentation (often preceded by noise reduction)
- Determines how different pixels are from their neighbours: abrupt changes in brightness are interpreted as the edges of objects
- The aim is to provide evidence for the automatic delineation of objects in a scene, which are assumed to be characterised by their edges



Edge Detection in Medical Images (source [1])

[1] https://www.researchgate.net/publication/325750965_An_Overview_of_Segmentation_Algorithms_for_the_Analysis_of_Anomalies_on_Medical_Images

First order derivatives

General pipeline:

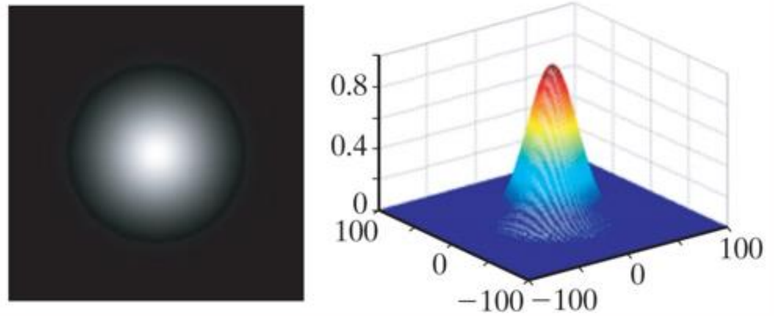
- Smoothing (to reduce noise) -> Derivative (so as noise is not accentuated)
- *Most differentiating kernels are built combining these two operations*

First order derivatives:

- 1D Gaussian derivative kernels
- 2D Gaussian derivative kernels: Image gradients

$$\nabla f(x, y) = \left[\frac{\partial f}{\partial x} \quad \frac{\partial f}{\partial y}\right]^{\mathsf{T}}$$

The Gaussian kernel



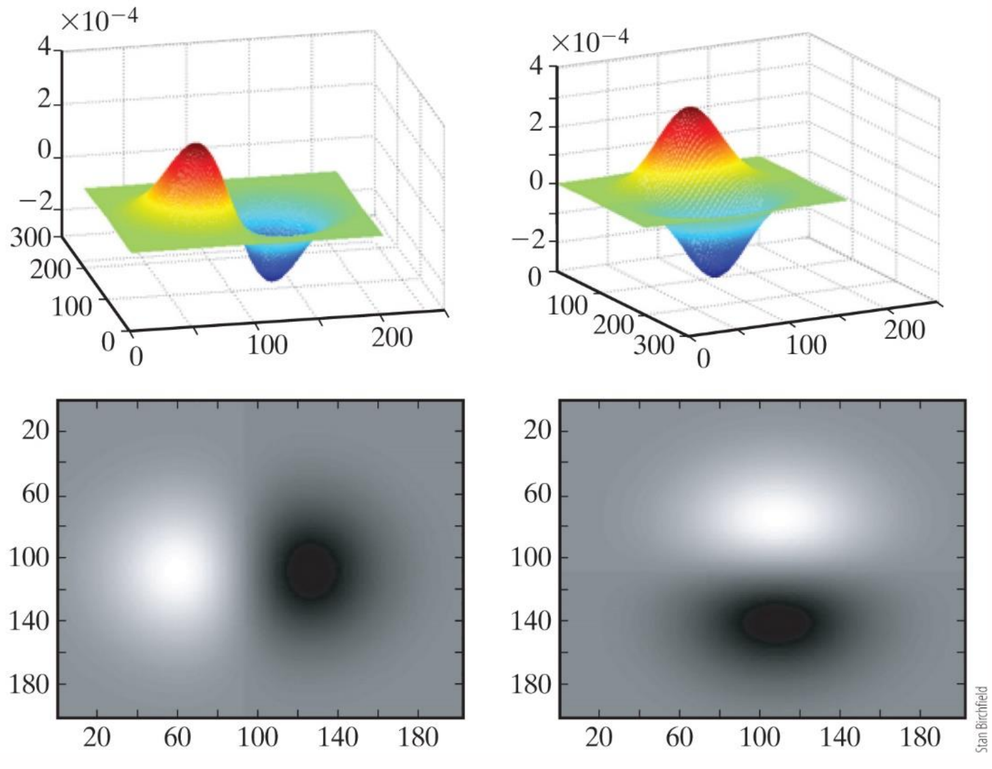


Figure 5.10 The 2D Gaussian partial derivatives in the *x* and *y* directions, shown as 3D plots (top) and images (bottom).

Stan Birchfield

First order derivatives

Prewitt operator:

- Simplest 2D differentiating kernel
- Obtained by convolving a 1D Gaussian derivative kernel with 1D box filter in the orthogonal direction

Box (Smoothing)

1D Gaussian (Differentiating)

$$Prewitt_x = \frac{1}{3}\begin{bmatrix}1\\1\\1\end{bmatrix} \circledast \frac{1}{2}\begin{bmatrix}1 & 0 & -1\end{bmatrix} = \frac{1}{6}\begin{bmatrix}1 & 0 & -1\\1 & 0 & -1\\1 & 0 & -1\end{bmatrix}$$

$$Prewitt_y = \frac{1}{3}\begin{bmatrix}1 & 1 & 1\end{bmatrix} \circledast \frac{1}{2}\begin{bmatrix}1\\0\\-1\end{bmatrix} = \frac{1}{6}\begin{bmatrix}1 & 1 & 1\\0 & 0 & 0\\-1 & -1 & -1\end{bmatrix}$$

Sobel operator:

- more robust, as it uses the Gaussian ($\sigma^2 = 0.5$) for the smoothing kernel:

Scharr operator:

- similar to Sobel, but with a smaller

variance ⟦(σ)⟧^2 =0.375) in the smoothing kernel

1D Gaussian (Smoothing)

1D Gaussian (Differentiating)

$$Sobel_x = gauss_{0.5}(y) \circledast \dot{g}auss_{0.5}(x) = \frac{1}{4}\begin{bmatrix}1\\2\\1\end{bmatrix} \circledast \frac{1}{2}\begin{bmatrix}1 & 0 & -1\end{bmatrix} = \frac{1}{8}\begin{bmatrix}1 & 0 & -1\\2 & 0 & -2\\1 & 0 & 1\end{bmatrix}$$

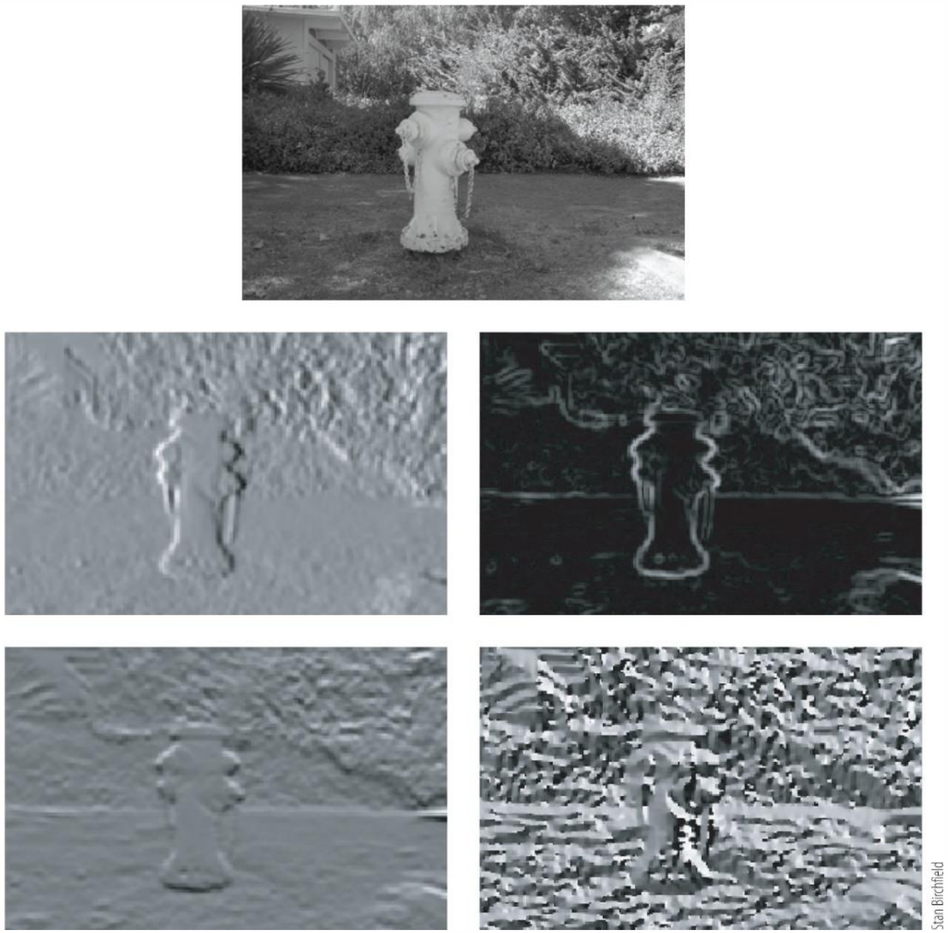
$$Sobel_y = gauss_{0.5}(x) \circledast \dot{g}auss_{0.5}(y) = \frac{1}{4}\begin{bmatrix}1 & 2 & 1\end{bmatrix} \circledast \frac{1}{2}\begin{bmatrix}1\\0\\-1\end{bmatrix} = \frac{1}{8}\begin{bmatrix}1 & 2 & 1\\0 & 0 & 0\\-1 & -2 & -1\end{bmatrix}$$

First order derivatives

Example:

Calculate ‘magnitude images’ of the directional image gradients.

Figure 5.11 Top: An image. Left: The partial derivatives of the image in the *x* and *y* directions, which together form the two components of the gradient of the image. Right: The magnitude and phase of the gradient.



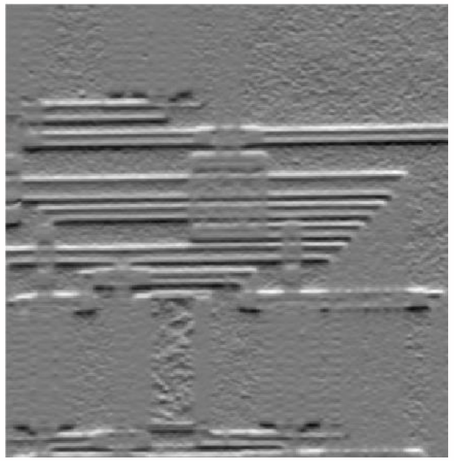
Stan Birchfield

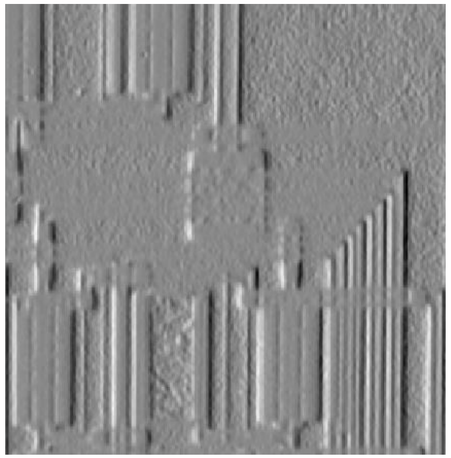
First order derivatives

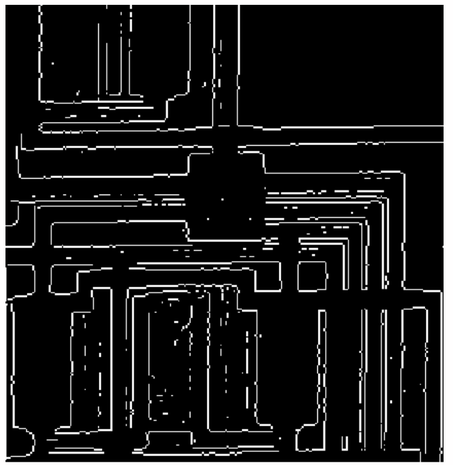
Example:

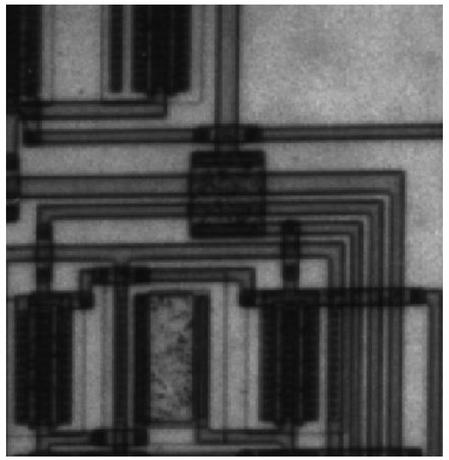
Calculate edges by thresholding ‘magnitude images’ of the directional image gradients.

Prewitt Operator



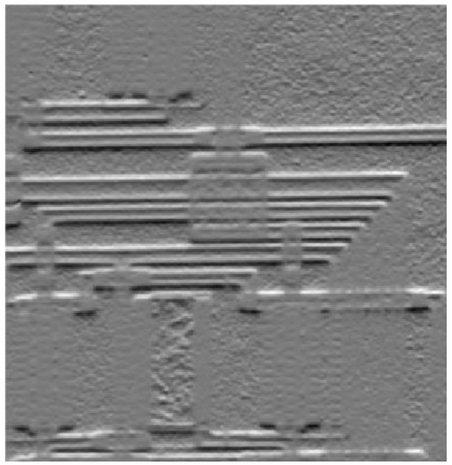




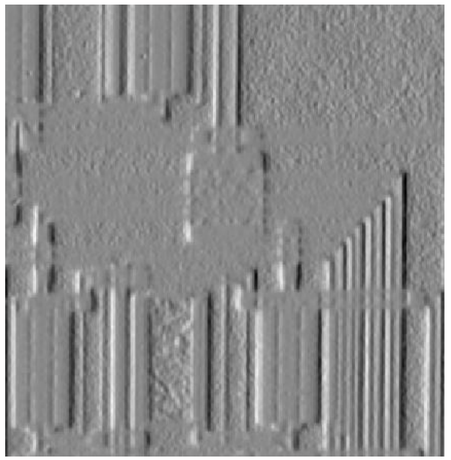


Input Grayscale image

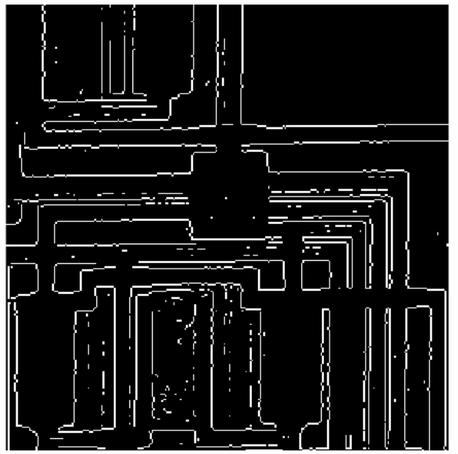
Sobel Operator



Vertical gradient



Horizontal gradient



Thresholded gradient magnitude

Second order derivatives

For a function (or image) of two variables, the second-derivative in the x and y directions can be obtained by convolving with the appropriately oriented **second-derivative kernel**

$$\frac{\partial^2 I(x,y)}{\partial x^2} = I(x,y) \circledast \begin{bmatrix} 1 & -2 & 1 \end{bmatrix}$$

$$\frac{\partial^2 I(x,y)}{\partial y^2} = I(x,y) \circledast \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$$

What makes this a 2nd order derivative?

This can be seen in 1D by convolving the function with the noncentralized difference operator, then convolving the result again with the same operator:

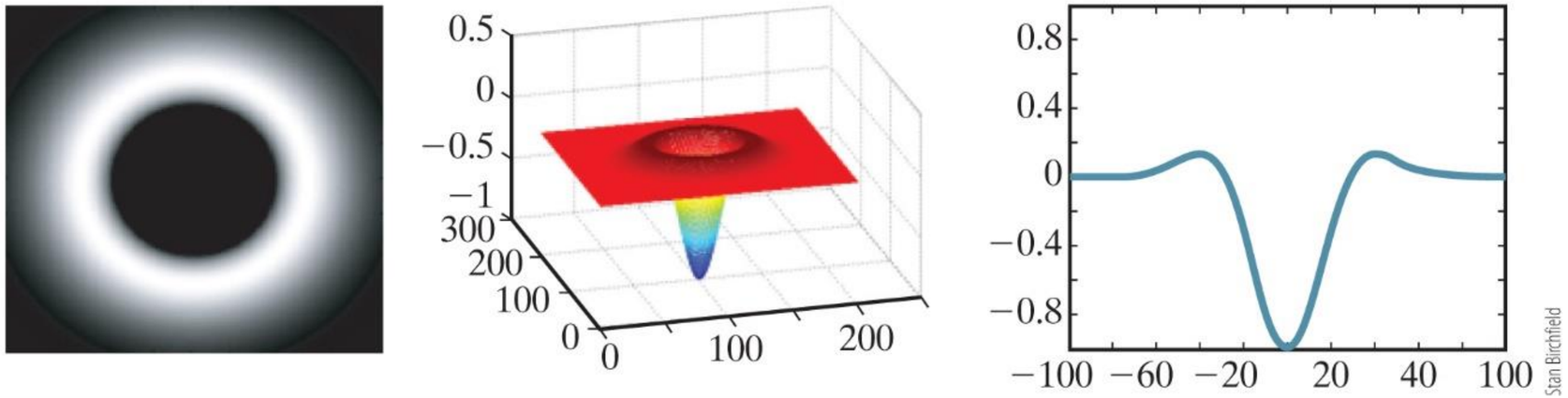
$$(f(x) \circledast [1 \quad -1]) \circledast [1 \quad -1] = f(x) \circledast ([1 \quad -1] \circledast [1 \quad -1]) = f(x) \circledast [1 \quad -2 \quad 1]$$

Second order derivatives

NOTE: These 2D derivatives are not isotropic or symmetric!

Is there any isotropic 2D derivative operators?

Figure 5.13 The Laplacian of Gaussian, presented as an image (left), a 2D plot (middle), and a 1D slice through the 2D plot (right). The center-surround nature of the operator is evident in the image, while the inverted Mexican hat shape is evident in the plots.



Second order derivatives

Laplacian of Gaussian (LoG) operator (isotropic, symmetric):

- Laplacian alone is defined as sum 2nd derivatives along the two axes:

$$\nabla^2 I = \nabla \cdot \nabla I = \begin{bmatrix} \frac{\partial}{\partial x} & \frac{\partial}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial I}{\partial x} & \frac{\partial I}{\partial y} \end{bmatrix}^{\mathsf{T}} = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \qquad (5.80)$$

- Laplacian, being a 2nd derivative, is highly sensitive to noise, therefore, preceded by Gaussian smoothing.
- *Remember? Most differentiating kernels are built combining two operations of smoothing and derivatives.*
- LoG is built by taking 2nd derivative of Gaussian

1D Gaussian (Smoothing)

1D Laplacian/2nd derivative (Differentiating)

$$\frac{\partial^2 Gauss_{0.25}}{\partial x^2} = \begin{bmatrix} 1 & -2 & 1 \end{bmatrix} \circledast \frac{1}{8} \begin{bmatrix} 1 \\ 6 \\ 1 \end{bmatrix} = \frac{1}{8} \begin{bmatrix} 1 & -2 & 1 \\ 6 & -12 & 6 \\ 1 & -2 & 1 \end{bmatrix} \qquad (5.87)$$

$$\frac{\partial^2 Gauss_{0.25}}{\partial y^2} = \frac{1}{8} \begin{bmatrix} 1 & 6 & 1 \end{bmatrix} \circledast \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} = \frac{1}{8} \begin{bmatrix} 1 & 6 & 1 \\ -2 & -12 & -2 \\ 1 & 6 & 1 \end{bmatrix} \qquad (5.88)$$

$$LoG_{0.25} = \frac{\partial^2 Gauss_{0.25}}{\partial x^2} + \frac{\partial^2 Gauss_{0.25}}{\partial y^2} = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \\ 2 & -12 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad (5.89)$$

Second order derivatives

Ollscoil na Gaillimhe
University of Galway

Laplacian of Gaussian (LoG) operator:

- Due to its smoothing function, and to the fact that it responds to both increasing and decreasing image intensity, the LoG produces edge maps with multiple responses to each 'true' edge
- characterised by multiple-pixel wide, inexact edges.

Figure 5.14 A 2304 × 1728 image, and the result of convolving with an isotropic LoG with different standard deviations.

Second order derivatives

Ollscoil na Gaillimhe
University of Galway

Difference of Gaussian (DoG) operator

- Very similar to LoG

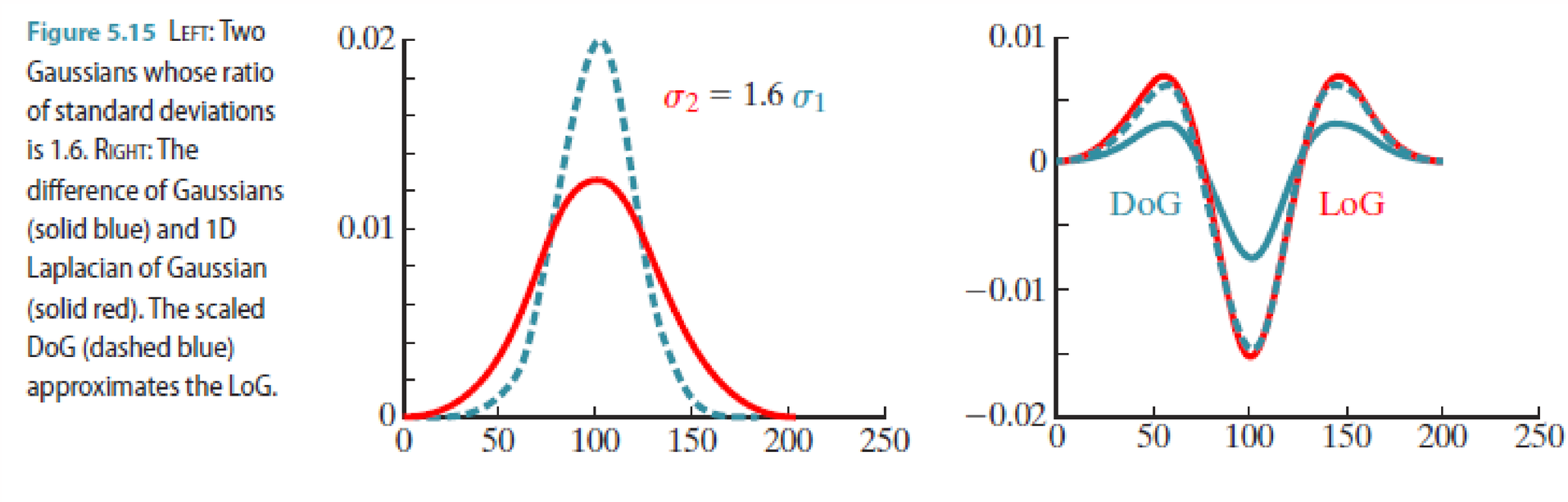


Figure 5.15 LEFT: Two Gaussians whose ratio of standard deviations is 1.6. RIGHT: The difference of Gaussians (solid blue) and 1D Laplacian of Gaussian (solid red). The scaled DoG (dashed blue) approximates the LoG.

Lecture 4: Image Filtering

Ollscoil na Gaillimhe
University of Galway

- Last Lecture:
  - Introduction to image processing
  - Point and geometric transformations
  - Spatial filtering (part 1 of 2)
    - 2D Convolution
    - Smoothing filters

- Today:
  - Image Filtering in Spatial Domain (part 2 of 2)
    - Differentiating filters: 1st order and 2nd order

  - Image Filtering in Frequency Domain
    - What is a frequency domain? (some Digital Signal Processing!)
    - How are images represented as (2D) frequencies? 2D Fourier Transform
    - Low-pass and high-pass filters

Image Filtering in Frequency Domain

Ollscoil na Gaillimhe
University of Galway

What is a frequency domain?

Any signal, discrete or continuous, periodic or non-periodic, can be represented as a sum of sinusoidal waves of different frequencies and phases which constitute the frequency domain representation of that signal.

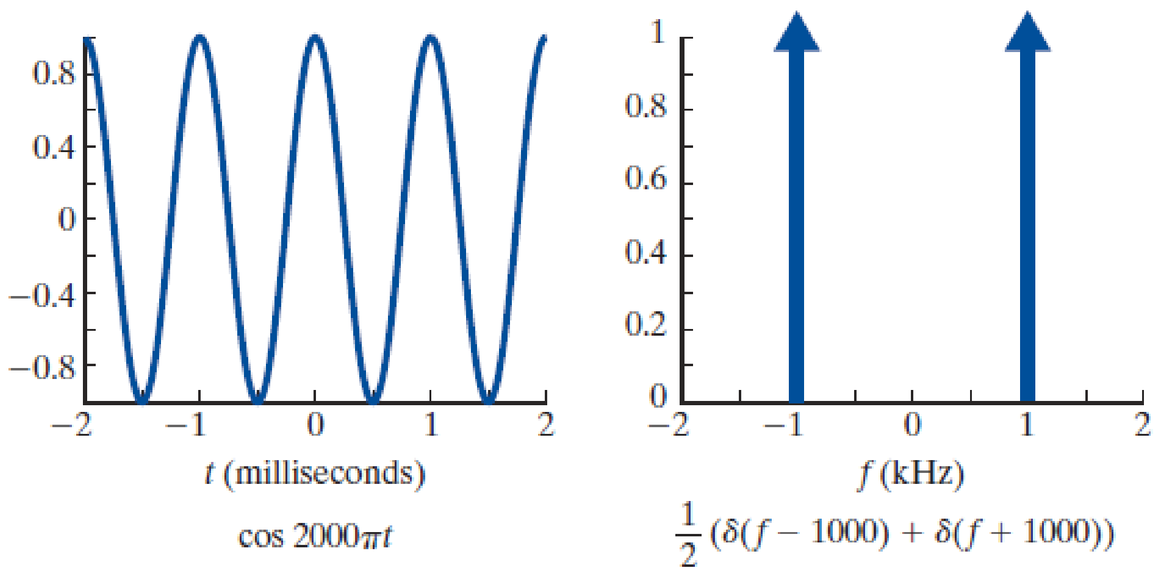


Figure 6.1 A continuous time-domain signal (left) and its Fourier transform (right). The latter reveals that the signal is a pure sinusoid with frequency 1000 Hz, since it contains two infinite spikes (Dirac deltas) at $f = 1$ kHz and $f = -1$ kHz. Note that the multiplicative factor $\frac{1}{2}$ has no effect on the display. See the text for an explanation of the negative frequency.

Image Filtering in Frequency Domain

OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

What is a frequency domain?

Let us build a signal from a combination of sinusoidal waveforms.

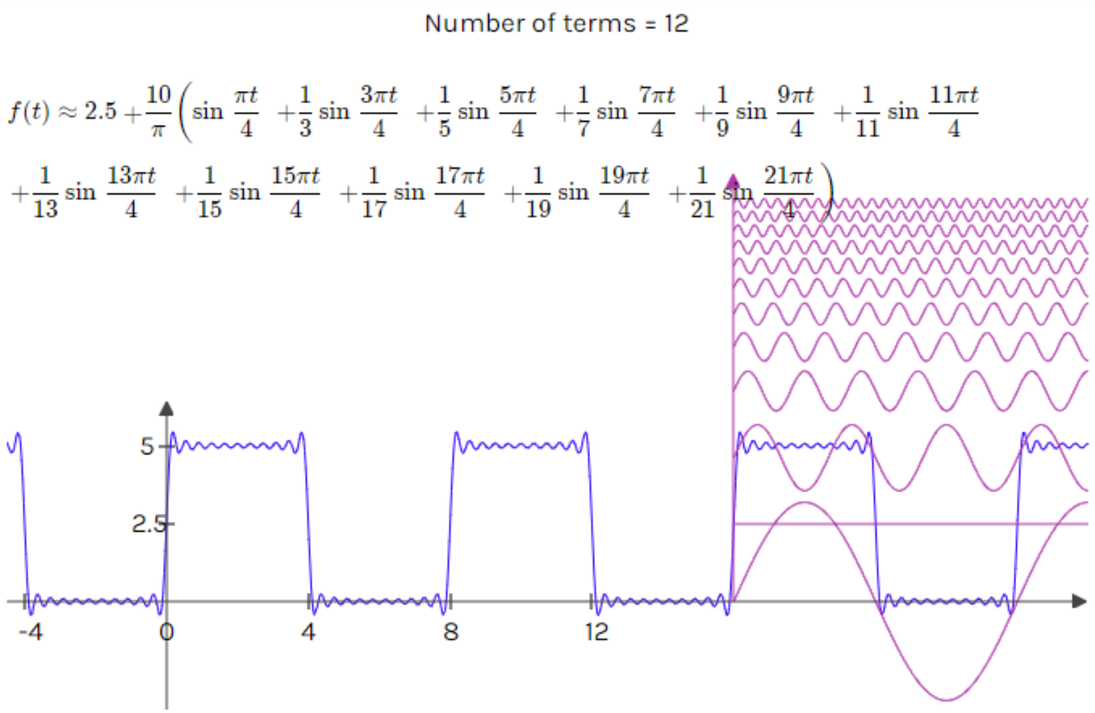


Image reproduced from and interactive online tool
(https://www.intmath.com/fourier-series/fourier-graph-applet.php)

Image Filtering in Frequency Domain

Ollscoil na Gaillimhe
University of Galway

Images (2D discrete signals) in the frequency domain

Figure 6.11 Multiplying the image by $(-1)^{x+y}$ prior to taking the DFT causes the result to be shifted so that the DC component is in the center. On the right is shown the logarithm of the magnitude of the DFT of the post-multiplied image.

How do we obtain a frequency domain representation of an image? Fourier Analysis

Fourier Transform: Fourier analysis of discrete time signals which are of finite duration (image grid) and assumed periodic. Both signals and frequencies are discrete and finite.

Fourier Transform

Forward Fourier transform G(f): the integration of the signal after first multiplying by a certain complex exponential:

$$G(f) \equiv \mathcal{F}\{g\} \equiv \int_{-\infty}^{\infty} g(t) e^{-j2\pi ft}\, dt$$

If t is measured in seconds, then f is measured in inverse seconds, also known as hertz.

Inverse Fourier transform: defined in exactly the same way as the forward Fourier transform except for the sign in the exponent, and the fact that the integral is computed over frequency rather than over time:

$$g(t) = \mathcal{F}^{-1}\{G\} \equiv \int_{-\infty}^{\infty} G(f) e^{j2\pi ft}\, df$$

Discrete Fourier Transform

Discrete Fourier transform:

Let g(x) be a 1D discrete signal with w samples. The DFT of g is defined as the summation of the signal after multiplying by a certain complex exponential, where both x and k are discrete integers.

Discrete

$$G(k) = \mathcal{F}\{g(x)\} = \sum_{x=0}^{w-1} g(x) e^{-j2\pi kx/w}$$

Discrete Fourier Transform

Continuous

$$G(f) \equiv \mathcal{F}\{g\} \equiv \int_{-\infty}^{\infty} g(t) e^{-j2\pi ft}\, dt$$

Fourier Transform

All modern implementations of the DFT use some variation of the FFT (Fast Fourier Transform) algorithm.

Discrete Fourier Transform

The 2D Discrete Fourier transform is a natural extension of the 1D case:

Replace the single frequency *k* with two frequencies in the two directions, k_x and k_y, so that kx/w becomes $k_x x/w + k_y y/h$.

$$G(k_x, k_y) = \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} g(x, y) e^{-j2\pi \mathbf{x}^T \mathbf{f}} \qquad \text{(forward DFT)}$$

$$g(x, y) = \frac{1}{wh} \sum_{k_x=0}^{w-1} \sum_{k_y=0}^{h-1} G(k_x, k_y) e^{j2\pi \mathbf{x}^T \mathbf{f}} \qquad \text{(inverse DFT)}$$

Fun Fact: Where do you use this in your daily life?

Frequency Domain Filtering

- Convolution in spatial domain is equivalent to multiplication in the frequency domain.
- To convolve two signals, take their Fourier Transforms, multiply and take the inverse Fourier Transform to get the result.

$$f'(x) = f(x) \circledast g(x) = \mathcal{F}^{-1}\{\mathcal{F}\{f(x)\} \cdot \mathcal{F}\{g(x)\}\} \qquad (5.8)$$

- **Efficiency**: Since multiplication is less expensive than convolution, this trick saves a lot of computational power when convolution kernels are big. (Why is this important even in the current age of almost limitless computation power?)
- Filtering is used primarily for two applications:
  - **Restoration:** the goal is to remove the effects of noise that has degraded the image quality from its original condition.
  - **Enhancement:** involves accentuating or sharpening features to make the image more useful, going beyond simply a pure, noise-free image.

Frequency Domain Filtering

A Low-pass Filter allows low frequencies to pass through while attenuating high frequencies.

The **ideal lowpass filter**, also known as the box filter, perfectly passes all frequencies below a certain cutoff, while perfectly attenuating all frequencies above the cutoff.

$$|H_{ilp}(f)| = \begin{cases} 1 & \text{if } f \leq f_c \\ 0 & \text{otherwise} \end{cases}$$

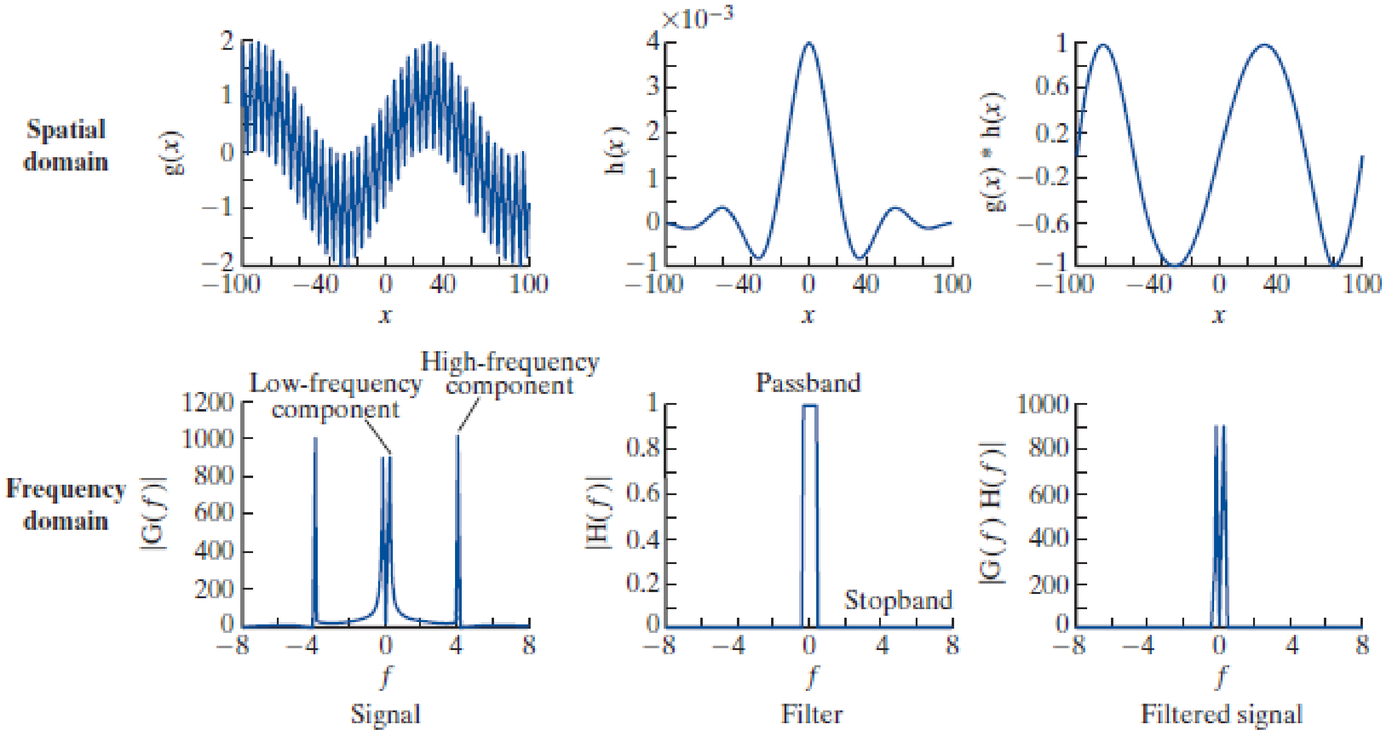


Figure 6.13 The function $\sin(f_1 x) + \sin(f_2 x)$ filtered by an ideal lowpass filter. In the frequency domain, the Fourier transform of the signal is multiplied by a box function. Equivalently, in the spatial domain, the signal is convolved with a sinc function. In this example the filter successfully removes the high-frequency component from the signal, leaving only the low-frequency component.

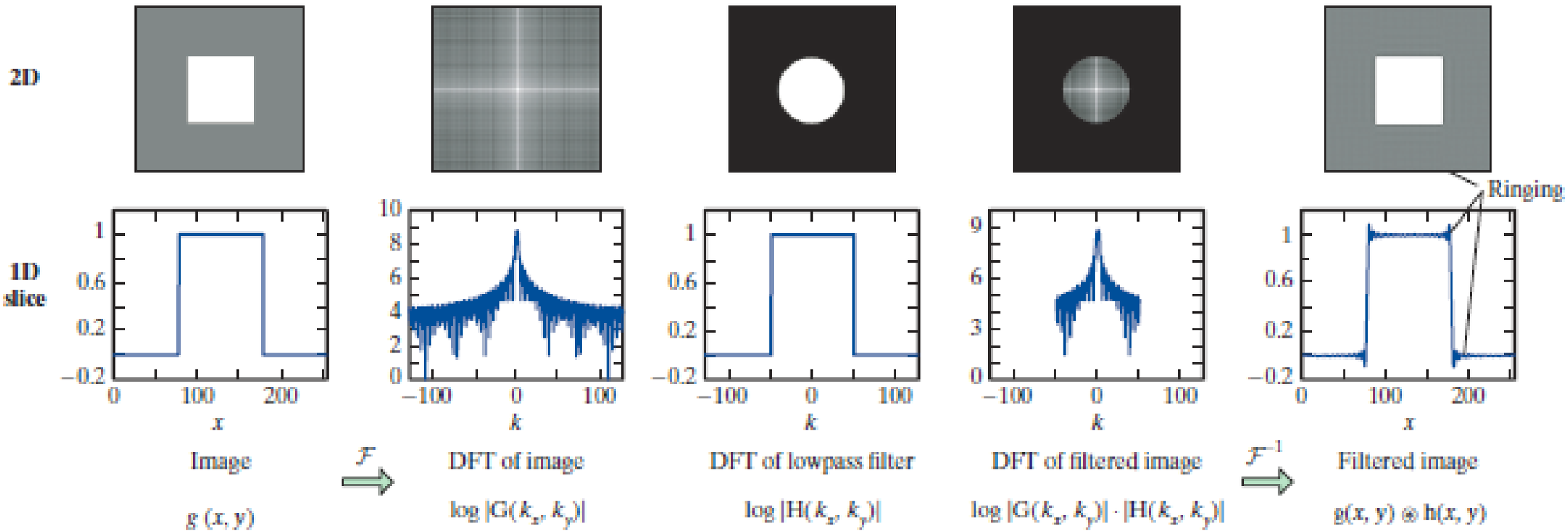
But is a box filter really an IDEAL filter?

Frequency Domain Filtering

Ollscoil na Gaillimhe
University of Galway

Filtering higher frequencies can result in image artefacts.

Figure 6.14 The process of frequency-domain filtering. From left to right: The DFT of the image is computed and multiplied by the frequency-domain filter, followed by the inverse DFT to yield the filtered image. Notice in this example that the ideal lowpass filter causes significant ringing in the output.



Frequency Domain Filtering

Gaussian Low-pass Filtering $|H_{glp}(f)| = e^{-f/2f_c^2}$

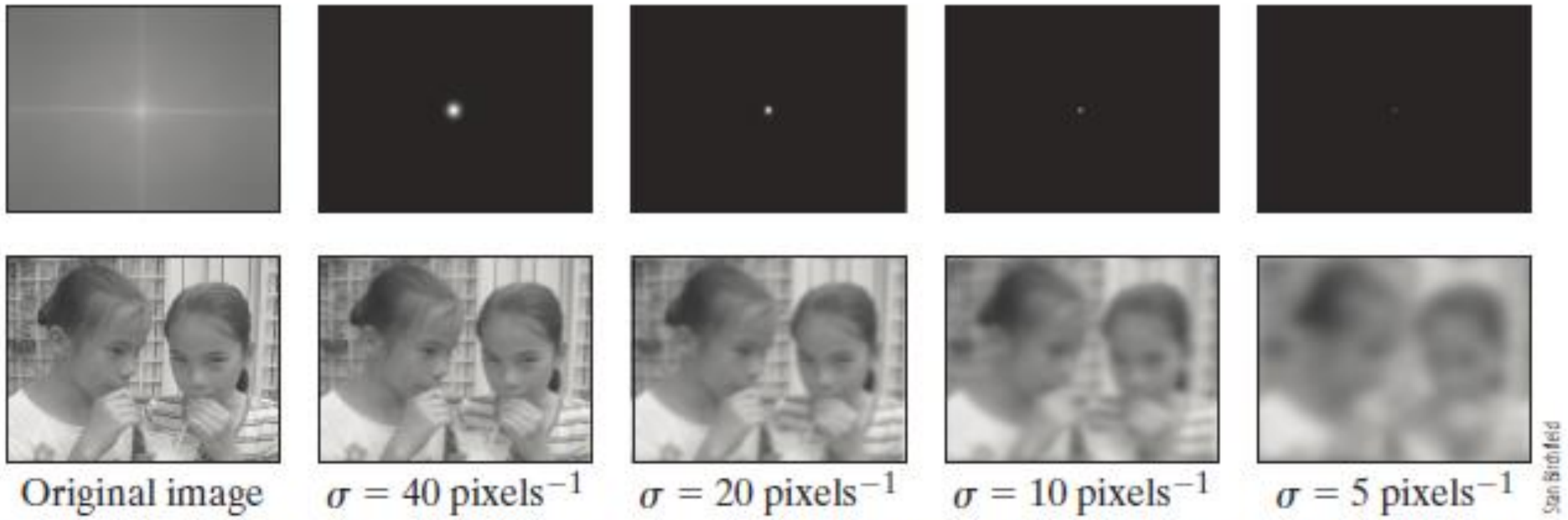


Figure 6.16 An image, and the result of Gaussian low-pass filtering in the frequency domain with different variances. The top row shows the DFT of the image and the magnitude of the frequency response of each filter. The smoothed images are the inverse DFT of the multiplication of the image DFT with the various filter frequency responses. Note that a large variance in the frequency domain yields less smoothing, whereas a small variance yields more smoothing.

Frequency Domain Filtering

Ollscoil na Gaillimhe
University of Galway

High-pass filters: obtained via negating an allpass filter with the magnitude of a lowpass filter.

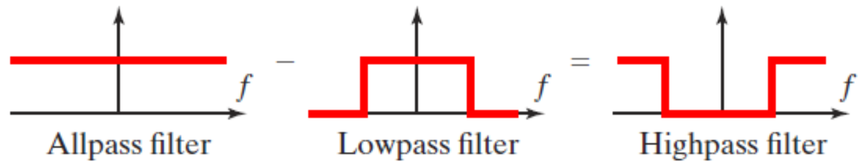


Figure 6.18 A highpass filter is the allpass filter minus a lowpass filter

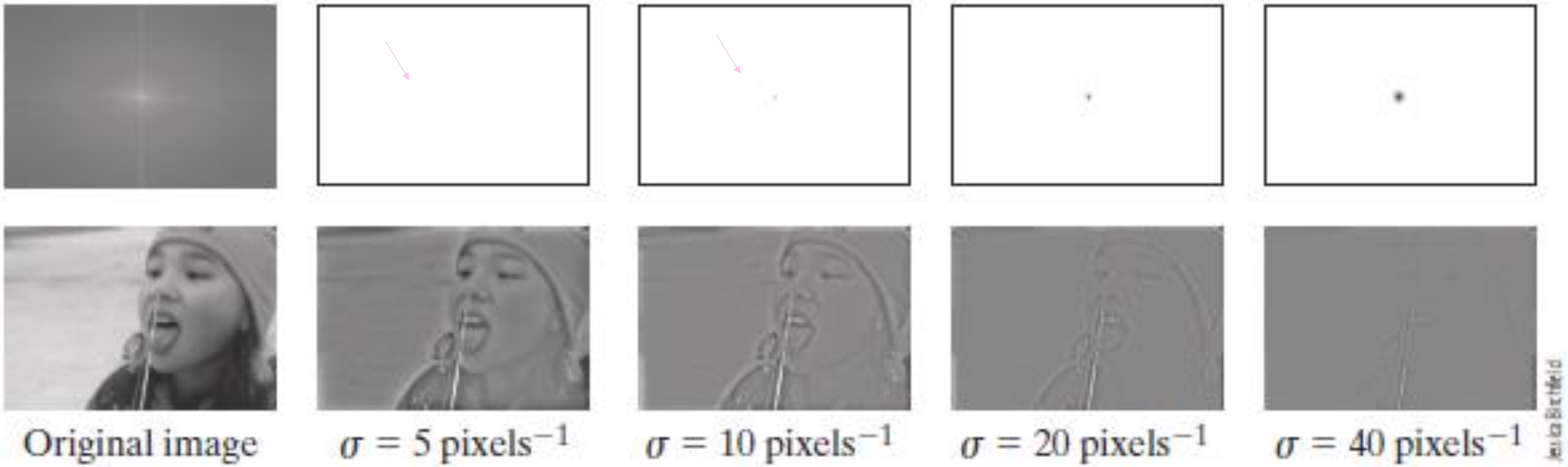


Figure 6.19 An image, and the result of Gaussian high-pass filtering in the frequency domain with different variances. The top row shows the DFT of the image and the magnitude of the frequency response of each filter. Bright values indicate frequencies that are passed, whereas dark values indicate frequencies that are attenuated.

Next Time

Ollscoil na Gaillimhe
University of Galway

- Binary image processing (image morphology)

Ollscoil na Gaillimhe
University of Galway

Thank *you*

University
*of*Galway.ie